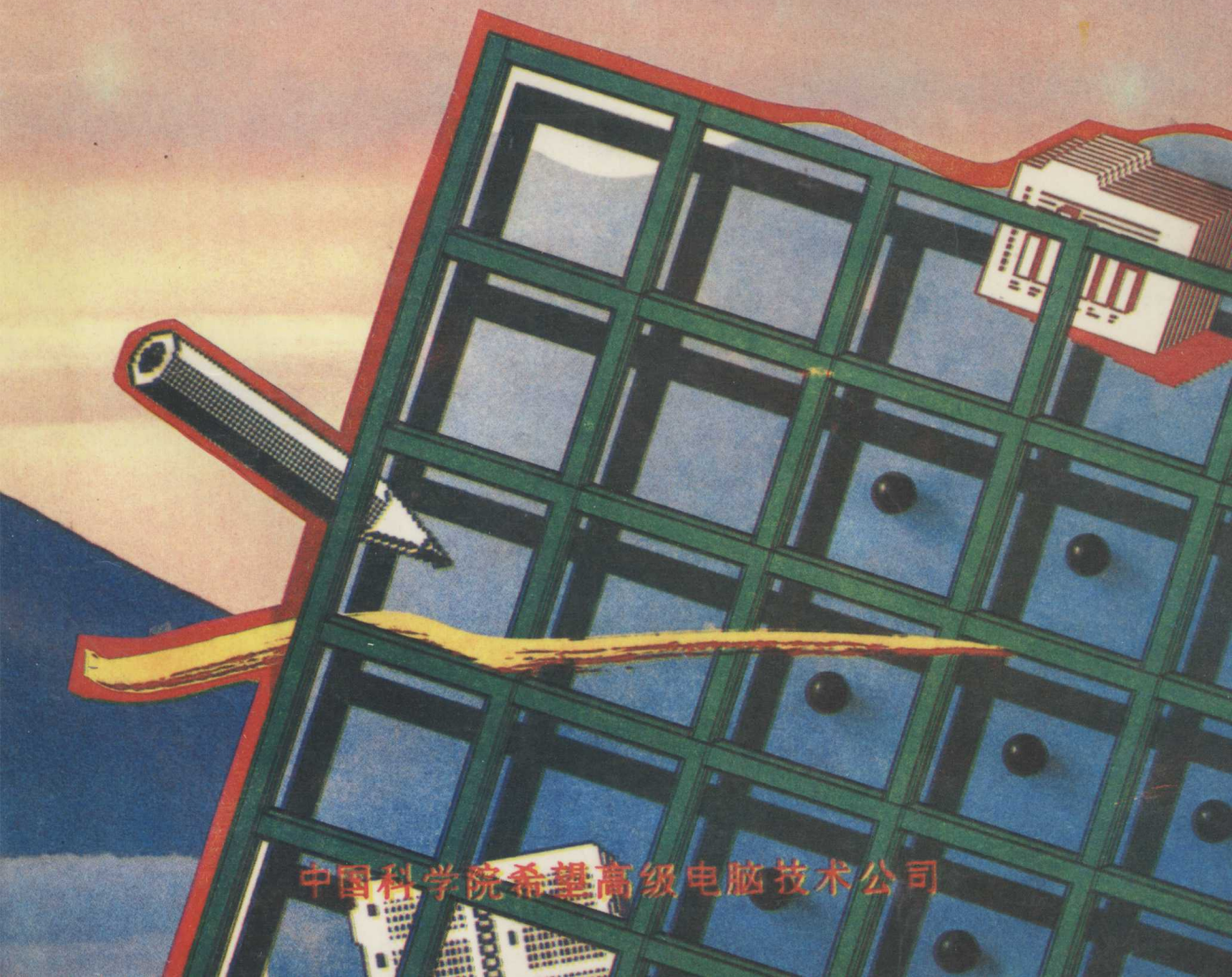
适用于IBM PCAT286.386.486

3.0

Microsoft Windows

最新窗口软件程序员参考手册

（上册）



中国科学院希望高级电脑技术公司

Microsoft Windows (3.0 版)

最新窗口软件程序员参考手册

(上册)

刘京涌　胡达　晓飞　　编译

中国科学院希望高级电脑技术公司

一九九一年四月

目　录

简　介

应用程序接口

本手册描述 Microsoft Windows 的应用程序接口(API)。API 包含函数库、消息、数据结构、数据类型、语句和文件。应用程序开发者用这些来设计在 Windows 中运行的程序。

API 可以认为是一套工具，适当地使用它能设计出可移植到各种计算机上的 Windows 应用程序。

Windows 的特点

Windows 应用程序可以利用 API 提供的许多特点。这些特点有:

- ■共享显示、内存、键盘、鼠标和系统定时器。
- ■与别的应用程序进行数据交换。
- ■与设备无关的图形。
- ■多任务
- ■动态链接

Windows 允许同时运行于系统上的多个应用程序共享硬件资源。应用程序的设计者不必为完成这项任务而去编写特定的代码。

Windows 的另一特点是用剪裁板(dipboard)来作为应用程序间交换数据的场所。应用程序间交换的信息可以是文本、位图(bitmaps)的形式，也可以是图形操作。Windows 提供了许多函数和消息来管理通过剪裁板进行的信息传送。这些函数及相应的消息是窗口管理接口中的一部分。窗口管理接口是 API 许多库中的一个。

Windows 中有用来编写与设备无关的图形操作的函数。这些函数产生的输出与光栅显示。各种型号的打印机以及许多点阵设备(绘图仪)的输出是兼容的。这些函数放在 API 的第二个库——图形设备接口中。

Windows 提供多任务，即许多应用程序能同时运行。影响多任务和内存管理的函数在 API 的第三个库——系统服务接口中。

由于 DOS 的内存限制，所以尽可能地使应用程序紧凑是很重要的。Windows 通过动态链接和使用可丢弃的代码来实现紧凑，即允许应用程序在运行期间装入执行函数库的子集。不管有多少应用程序在使用一个库，也只需要这个库的一个备份。

窗口管理接口

窗口管理接口中含有 Windows 应用程序中最基本成分的函数如创建、移动、改变窗口的函数。窗口是一个矩形区域，含有用户输入的图形表示、输入选择和系统输出。

Windows 是一个菜单驱动的环境。在一个应用程序中用户表达选择的主要手段是菜单。那些创建菜单、改变菜单的内容、得到菜单选项状态的函数也是窗口管理接口中的一部分。

窗口管理接口也有产生系统输出的函数。这种输出的一个例子就是对话框，应用程序可以用它来要求用户输入、显示信息。

窗口管理接口也含有消息及处理消息的函数。消息(message)是一种特殊的数据结构，用来表示有关应用程序内发生的变化的信息。这些变化包括键盘、鼠标和定时器的活动。信息的请求或应用程序应该执行的操作。

窗口管理者接口函数组

窗口管理接口中有以下函数组

- ■消息函数
- ■信息函数
- ■窗口创建函数
- ■系统函数
- ■显示和移动函数
- ■剪裁板函数
- ■错误函数
- ■输入函数
- ■插入符函数
- ■硬件函数
- ■光标函数
- ■绘画函数
- ■钩函数
- ■对话函数
- ■特性函数
- ■滚屏函数
- ■矩形函数
- ■菜单函数

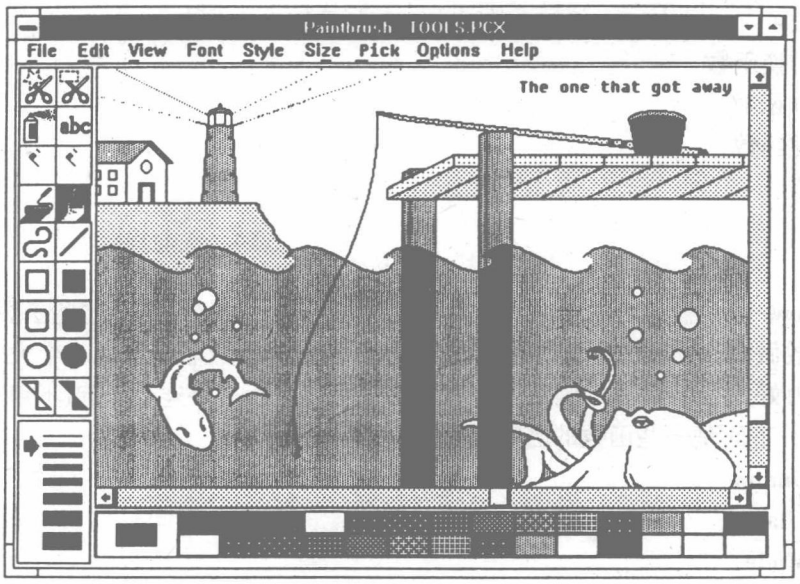
图形设备接口

图形设备接口(GDI)是一个有关在 Windows 应用程序中执行与设备无关的图形操作的函数库。这些函数能在许多不同的输出设备上产生一条、文本和位图的输出。GDI 允许应用程序为特定的输出设备产生笔、刷、钻字和位图的输出。下面这张图就是由文本、线条和绘图输出组成的，它是由用 GDI 函数库编写的应用程序 Microsoft Windows Paintbrush 产生的。

图形设备接口函数组

在 GDI 中你可以找到以下函数

- ■设备环境函数
- ■椭圆和多边形函数
- ■画图工具函数
- ■位图函数
- ■画图属性函数
- ■文本函数
- ■映射函数



Text, Line and Bitmap Output

- ■字体函数库
- ■坐标函数库
- ■元文件函数
- ■区域函数
- ■打印机检码函数
- ■剪裁函数
- ■环境函数
- ■线输出函数
- ■系统函数

系统服务接口

系统服务接口中的函数可以完成以下功能：在模型中获取代码和数据，申请并管理内存(局部的和全局的)，管理任务，装入程序资源、所字符串从一字符集翻译到另一字符集上，改变 Windows 初始化文件，帮助调试系统，完成通过系统 I/O 端口的通讯，创建、打开文件，使用系统扬声器产生声音。

系统服务接口函数组

在系统服务接口中你可以找到以下函数组：

- ■模块管理函数
- ■初始化文件函数
- ■内存管理函数
- ■通讯函数
- ■任务函数
- ■声音函数
- ■资源管理函数

■实用函数
■字符串转换函数
■文件 I/O 函数
■原子管理函数
■系统函数

命名习惯

许多 Windows 函数被命名成动词——名词模式来帮助你记忆并熟悉这些函数。函数的名字指出了函数做什么(动词)和操作的目标(名词)。所有的函数名字都以一个大写字母开头。如果名字由好几个字组成，则每个字以大写开头并且所有字连在一起(不用空白或下横线来分开字)。下面是一些函数的名字例子:

■CreateWindow
■RegisterClass
■SetMapMode

参数名字

大多数参数和局部变量都有一小写字母前缀来表示参数的通常类型，小写字母后是一个或多个词来描述参数的内容。参数和变量使用的标准前缀定义如下:

前缀	意义
b	布尔值(非空值为真，零表示假)
c	字符(一字节值)
dw	无符号长(32位)整数
f	由位标志组成的16位整数
h	16位指针
l	长(32位)整数
lp	长(32位)指针
n	短(16位)整数
p	短(16位)指针
pt	由x,y坐标值组成的一个无符号32位整数。
rgb	由RGB色彩值组成的32位整数。
w	无符号短(16位)整数

如果说有给定小写字母前缀，则参数是一个短整数，它的名字是描述性的。

参数和变量名字的例子如下:

bloonic	ptXY
fAction	ngbCowr
hwnd	Height
lpString	X
nBytes	Width

pMsg Y

Windows 调用习惯

Windows 使用的调用格式与 Microsoft Pascal 使用的是一样的。这本手册中的调用格式参照了 Pascal 的调用格式。Pascal 的调用格式如下:

■参数压入堆栈的顺序按它们出现在函数调用中的顺序。

■从堆栈中恢复参数的代码在一被调用函数中(而不是在调用函数中)。

这个格式与别的语言(如 C)中使用的格式不一样。在 C 中，参数压入堆栈是序的，并且由调用函数负责从堆栈中恢复参数。

若用与 Pascal 调用格式不一致的语言(如 C)来开发 Windows 应用程序，你必须保证由 Windows 调用的函数使用了 Pascal 调用格式。在 C 中，这要求在函数说明时使用 APSCAL 关键词。

手册综述

本手册给 Windows 应用程序开发者提供了有关 Windows 函数、消息、数据类型、资源编译器语句、汇编语言宏和文件格式等的通用且详细的信息。它不想试图去解释如何创建一个 Windows 应用例程，而是给对 Windows 编程已经有了基本了解的读者提供 Windows API 每个组成部分的详细描述。

本书分成上、下两册，以下几节描述了每册的用途及组成。

上册

上册中包含的是有关描述 Windows 函数和消息的参考信息，由 6 章组成。

第一章“窗口管理者接口函数”把窗口管理者函数分成几个相关的组并简单描述了每个函数。这章还提供了每个函数组的信息，包括新名词的定义和对 Windows 特有模型的描述。设计这章是为了帮助那些新的 Windows 应用程序开发者和那些对 Windows 某个函数组有疑问的开发者。

第二章“图形设备接口函数”对在 Windows 环境中执行与设备无关的图形操作的函数进行了分类，并提供对函数简单描述，还解释了 Windows 图形接口中最重要的特点。

第三章“系统服务接口函数”对那些执行与窗口管理和产生图形输出不直接有关系的服务的各种实用函数进行了分类。

第四章“函数目录”含有以字母顺序排列的 Windows 函数名字者。每个函数的文档给出了语法、说明了函数的用途、列出了它的输入参数，还描述了它的返回值。对有些函数还给出开发者在使用这些函数时需要的额外信息。

第五章“消息综述”把消息按关系分成组并描述了每个消息。这章还提供了某些消息组的额外信息，包含新名词的定义和对 Windows 特有的模型的描述。设计本章是为了帮助新的 Windows 应用程序开发者和对某个 Windows 消息组有问题的开发者。

第六章“消息目录”给出了以字母顺序排列的 Windows 消息名字表。每个消息的文档说明了消息的用途，列出了它的输入参数，并描述了它的返回值(如果有的话)。对有些消息，还给出了开发者使用这些消息时需要的额外信息。

下册

下册含有 Windows API 别的组成成分的参考信息。它有 9 章和 5 个附录。

第七章“数据类型和结构”含有一张数据类型表和一张以字母顺序排列的 Windows 结构名表。

第八章“资源说明语句”描述了定义资源的语句。资源由资源编译器(Resource Compiler)加到应用程序的可执行文件中去。语句是根据功能组安排的。

第九章“文件格式”描述了五种其它的文件格式(五种类型的文件是：位图文件、图符资源文件、光标资源文件、剪裁板文件和元文件)。每个描述给出了通用的文件结构和文件各部分的具体信息。

第十章“模块定义语句”描述了在模块定义文件中用来定义应用程序内容和 Link 程序系统要求的语句。

第十一章“二元和三元光栅操作代码“”描述了用来线条输出和位图输出的光栅操作。

第十二章“打印机转义”列出了 Windows 中可使用的打印机转义函数。

第十三章“汇编语言宏综述”分类并简单描述了汇编语言宏，这些宏给函数和高级语言的段格式提供了简化的接口。

第十四章“汇编语言宏目录”以字母顺序列出了汇编语言宏，并对每个宏提供了详细的描述以及在程序中如何使用的一个到多个例子。

第十五章“WindowsDDE 协议定义”以字母顺序列举并描述了保证动态数据交换(DDE)约定的 Windows 消息。

附录 A“Windows 虚拟键码”列出了 Windows 虚拟键码的符号名和十六进制值，以及每个键的简单说明。

附录 B“RC 诊断信息”含有一张资源编译器(Resource Compiler)的错误信息表，并对每个信息进行了简单说明。

附录 C“Windows 调试信息”含有一张 Windows 调试信息表，并对每个信息进行了简单说明。

附录 D“字符表”给出了 ANSI 字符集和 IBMPC 扩展字符集。

附录 E“32 位内存管理 DDL”描述了在你的应用程序中如何完成 32 位内存 A 模型。

第一部分 Windows 函数

第一部分描述了 Windows 应用程序编程者接口(API)的主要函数。你可以把这些函数当作 C 或汇编语言程序的一部分，来编写使用 Windows 用户接口、图形和多任务性能的应用程序。

章节

第一章　窗口管理者接口函数

本章描述了 Microsoft Windows 的消息处理函数、窗口创建、移动、改变函数、系统输出函数。这些函数组成了窗口管理者接口。

本章论述下列话题:

- ■消息函数
- ■窗口创建函数
- ■显示、移动函数
- ■输入函数
- ■硬件函数
- ■绘画函数
- ■对话框函数
- ■滚屏函数
- ■菜单函数
- ■消息函数
- ■剪裁板函数
- ■错误函数
- ■插入符函数
- ■光标函数
- ■钩函数
- ■特性函数
- ■矩形函数

1.1　消息函数

消息函数读入并处理应用程序队中的 Windows 消息。消息代表了 Windows 应用程序的许多输入，它是一种含有消息标志符和消息参数的数据结构。消息参数的内容随消息的类型不同而改变。下表简单描述每个函数。

函数	说明
CallWindowProc	把消息信息传递给某一函数。
DispatchMessage	把消息传给某一窗口的窗口函数。
GretMessage	在特定范围的消息中得到一消息。
GetMessagePos	返回得到最后一条消息时的鼠标位置。
GetMessgaeTime	返回得到最后一条消息的时间。
InSendmessage	确定当前窗口函数是否正在处理通过调用SendMessage而传给它的消息。
PeekMessage	检查应用程序队列，把消息放在合适的地方。
PostAppMesage	把消息投寄给应用程序。
PostMessage	把消息放入应用程序队列中。

PostQuitMessage	把WM-QUIT消息投寄给应用程序。
ReplyMessage	对消息的回答。
SendMessage	创建一个不同大小的新的消息队列。
TranslateAccelerator	处理菜单命令的键盘加速键。
TranslateMDISysAccel	处理多个文件接口(MDI)子窗口命令加速键。
TranstateMessage	把虚击键消息翻译成字符消息。
WaitMessage	产生对别的应用程序的控制。
WinMain	作为Windows应用程序执行的入口点。

1.1.1 产生和处理消息

对每一次输入事件，例如用户移动鼠标或在键盘敲一个键，Windows 便产生一条消息。Windows 把这些输入消息收集在系统范围的一个队列中，然后把这些消息连同定时器消息和绘画消息放到应用程序的队列中去。应用程序队列(简称应用队列)是先进先出队列，属于某个应用程序。不过，定时器消息和绘画消息在队列中要保存到应用程序处理完别的消息为止。Windows 把某个应用程序的消息放到这个应用程序的队列中。应用程序使用函数 GetMessage 来读这些消息，使用 DispatchMessage 函数来发送他们到某个合适的窗口函数。

Windows 可以把某些消息直接送到应用程序的窗口函数而不必将他们放入应用队列。这样的消息叫非队列消息。通常非队列消息是指那些只影响窗口的消息。函数 SendMessage 可以直接把消息送到一个窗口。

例如，CreatWindows 指示 Windows 给应用程序的窗口函数发送“WM-CREATE”消息并且等待，直到这条消息被窗口函数处理完。Windows 把这条消息发送给函数而不是放入应用程序队列。

尽管大多数的消息是由 Windows 产生的，但应用程序也可以创建自己的消息，并把他们放到别的应用程序的应用程序队列中去。

应用程序通过使用 GetMessage 函数便可从队列中得到消息。GetMessage 函数从应用程序队列中寻找消息，如果应用程序队列中有消息的话，它就返回应用程序队列顶部的消息。如果应用程序队列为空，GetMessage 等待消息放入队列。在等待期间，GetMessage 把控制交给 Windows，允许别的应用程序得到控制权来处理他们的消息。

一旦主函数从队列中得到消息，它使用 DispatchMessage 函数把消息分发给窗口函数。DispatchMessage 函数指示 Windows 调用与这条消息有关的窗口函数，并把消息的内容作为函数参数传递给它。窗口函数便处理这些消息，完成对窗口的预定的变化。当窗口函数返回时，它把控制交还给主函数。主函数于是又从队列中取下一条消息。

注意：除非另外说明，Windows 可以以任意顺序发送消息。应用程序不应该按照某一顺序接收消息。

每当用户在键盘上敲一个键时，Windows 便产生一条虚键消息。虚键消息中含有虚键代码。虚键代码定义哪个键被敲，不定义被敲键的字符值。为得到字符值，主函数必须用 TranslateMessage 函数来翻译虚键消息。TranslateMessage 函数把含有相应字符值的消息放到应用程序队列中。这样消息才能被分发给窗口函数。